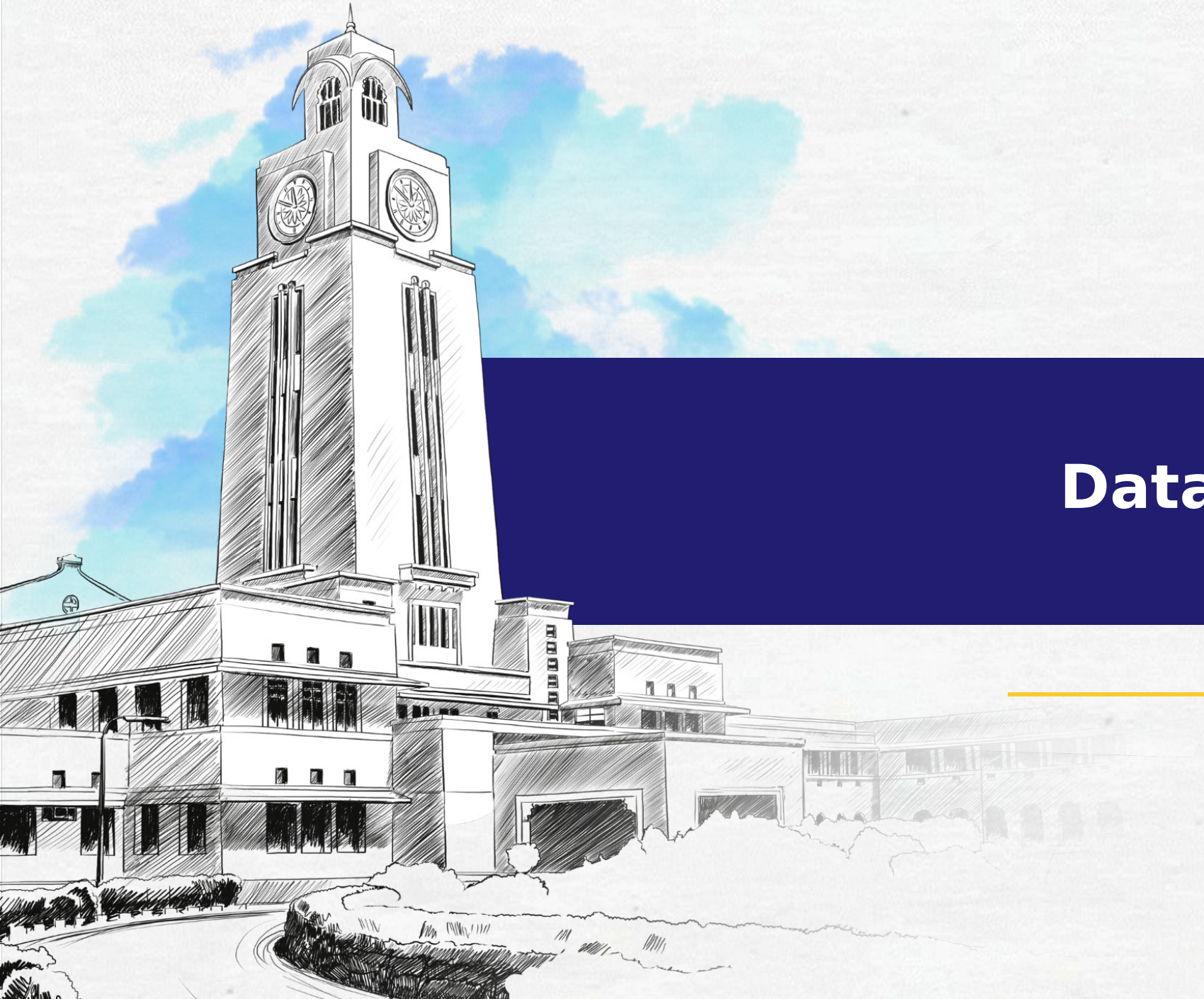


BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Data Representations

Sunil Bhutada

Agenda

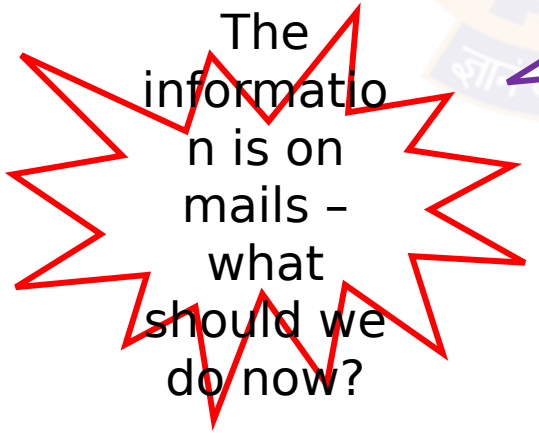
- Introduction
- Formats of Data
- Data Models
- Data Storage Layouts
- Data Serialization
- Data Processing Types

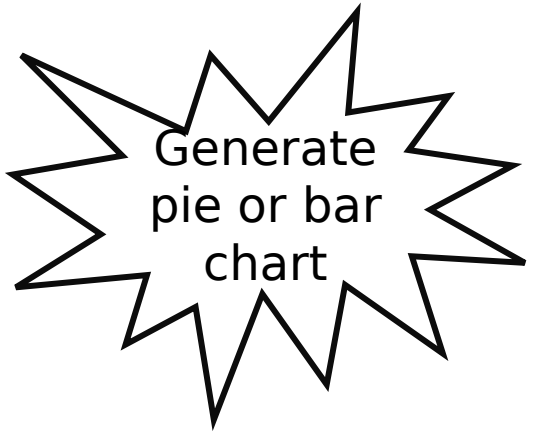
Abbreviations

NF	Normal Forms
BCNF	Boyce Code Normal Form
DB	DataBase
DBMS	DataBase Management System
OLTP	OnLine Transaction Processing
OLAP	OnLine Analytical Processing
JSON	JavaScript Object Notation
XML	eXtensible Markup Language
BSON	Binary JavaScript Object Notation
JSONB	JSON Binary
CSV	Comma Separated Values
SQL	Structured Query Language

Introduction

Have you heard these in your work?





What is Data Management?

- Those words that you heard are in a way connected to data management

- Data management is the practice of ingesting, processing, securing and storing an organization's
  data, where it is then utilized for strategic decision-making to improve business outcomes

- What are the key words?

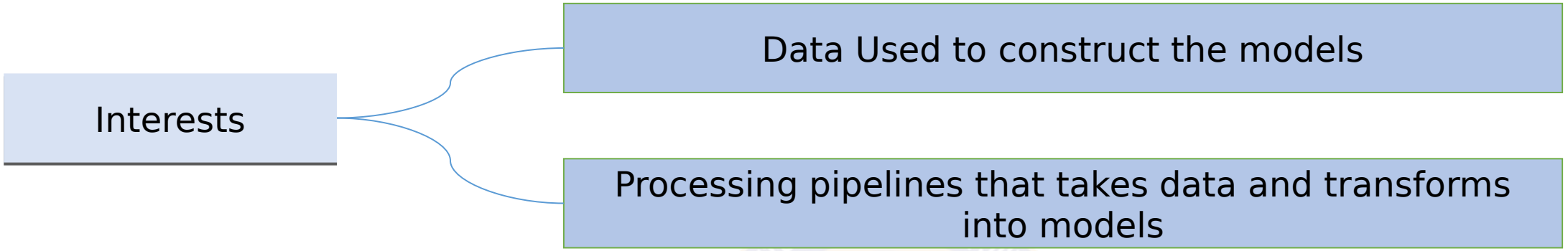
- Data management is the practice of **ingesting**, **processing**, **securing** and **storing** an organization's data, where it is then utilized for strategic decision-making to improve business outcomes

Why Data Management?

Growth of Big Data	Over the last decade, developments within hybrid cloud, artificial intelligence, the Internet of Things (IoT), and edge computing have led to the exponential growth of big data creating even more complexity for enterprises to manage
Increasing and changing priorities	A data management discipline within an organization has become an increasing priority as this growth has created significant challenges, such as data silos, security risks, and general bottlenecks to decision-making
Improve Business Outcomes	Need clean interactive data for business insights and improve the business outcomes

Why DM for ML????

Relationship between Data and ML Pipelines



What is a ML System?
Essentially a data processing pipeline? Yes

What is the purpose of ML System?
Extract usable and repeatable insights of data? Yes

How is this different from traditional log processing or analysis pipelines?
- ML systems' success are hard to measure
- Detection of failures are difficult
- Depends fully on underlying data (management) systems in terms of
  - ✔ Structure
  - ✔ Performance
  - ✔ Accuracy
  - ✔ Reliability

Formats of Data

Overview

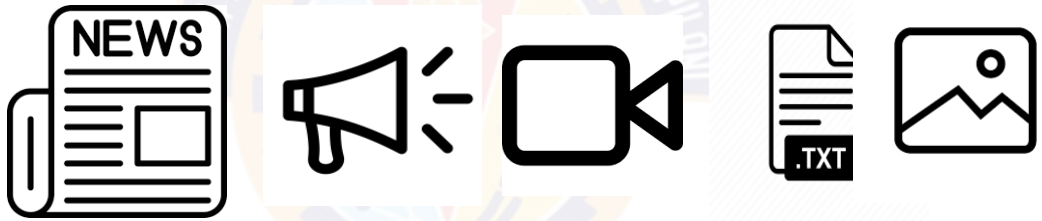
Types

- Structured

CourseID	CourseName	IsValid	Start Year	End Year
DSC	DataScience	Yes	2020	2035
AIML	Aritificial Intelligence and Machine Learning	Yes	2020	2040
DSCIO	DataScience with IOT	No	2020	2026
DSML	DataScience and Machine Learning	Yes	2020	2035

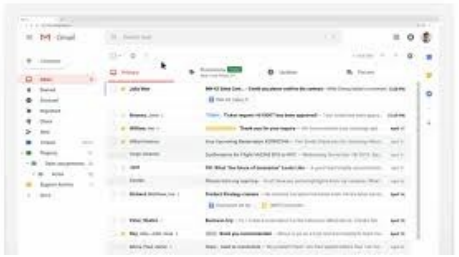
Follows a predefined schema or structured
(Schema : blueprint or structure that defines how data is organized, stored, and related. It specifies the rules, format, and constraints of the data)

- Unstructured



Doesn’t adhere to any schema

- Semi-Structured



Structured + Unstructured

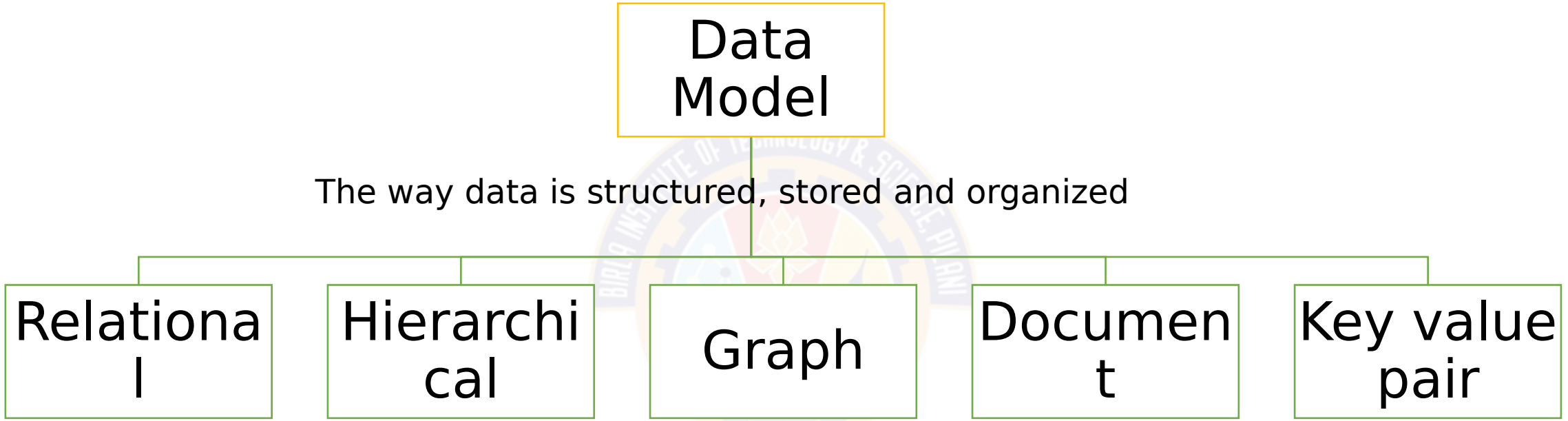
Structured vs Unstructured vs Semi-Structured

	Structured Data	Unstructured Data	Semi-Structured Data
What	A definite schema is available	No schema or no structure exists	A combination of both
Sample	The data that you capture in excel or csv	Hand written notes, news that you listen or see	Emails - where from, to, subject fields are available and content is your wish
Advantages	*Easy for data analysis *Faster to retrieve	*Wide range of storage options *Lot of data as well as lot of interpretations out of it *Adaptive to schema changes *Slower to retrieve information	*Evolving schema unlike structured where it is fixed and unstructured where it doesn't exist *Highly interoperable
Disadvantages	*Makes it complex if schema changes *Comitting to pre-defined schema is restricting *Growing types of sources not waranting schema *Definite storage format is requried	*Difficult for analysis	*Tricky to have a routine for validation and analysis

Home Work: What is **Quasi structure** and to which in the above category this is related to?

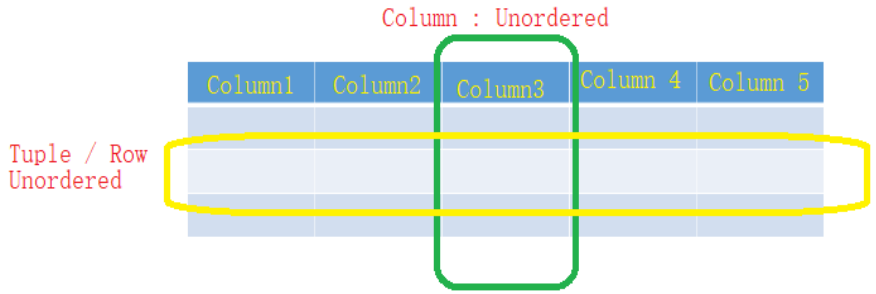
Data Models

Definition & Types



Relational Data Model

- Data is organized into relations; Each relation is a set of tu
- Table is visual representation of relation, each row of table makes up a tuple
- Relations are unordered – can shuffle order of rows or columns



StudentID	StudentName	CourseName	SubjectName
202512434	N1	DataScience	DataScience
202512435	N2	DataScience and Machine Learning	DataScience
202512436	N3	DataScience with IOT	DataScience (Python)
202512437	N4	Aritificial Intelligence and Machine Learning	Data Science
202512438	N5	Aritificial Intelligence and Machine Learning	DataSciences
202512439	N6	Aritificial Intelligence and Machine Learning	DataScience
202512434	N1	DataScience	Applied Maths(statistics & probabilty)
202512435	N2	DataScience and Machine Learning	Applied Maths
202512436	N3	DataScience with IOT	Applied Maths
202512437	N4	Aritificial Intelligence and Machine Learning	Applied Maths (P&S)
202512438	N5	Aritificial Intelligence and Machine Learning	Applied Maths
202512439	N6	Aritificial Intelligence and Machine Learning	Applied Maths

What is this?

Do you observe any problem?

Which is Better?

StudentID	StudentName	CourseName	SubjectName
202512434	N1	DataScience	DataScience
202512435	N2	DataScience and Machine Learning	DataScience
202512436	N3	DataScience with IOT	DataScience (Python)
202512437	N4	Aritificial Intelligence and Machine Learning	Data Science
202512438	N5	Aritificial Intelligence and Machine Learning	DataSciences
202512439	N6	Aritificial Intelligence and Machine Learning	DataScience
202512434	N1	DataScience	Applied Maths(statistics & probabilty)
202512435	N2	DataScience and Machine Learning	Applied Maths
202512436	N3	DataScience with IOT	Applied Maths
202512437	N4	Aritificial Intelligence and Machine Learning	Applied Maths (P&S)
202512438	N5	Aritificial Intelligence and Machine Learning	Applied Maths
202512439	N6	Aritificial Intelligence and Machine Learning	Applied Ma

Disadvantages:

- Data Integrity is not there
- Difficult to interpret
- Highly redundant values
- Not expandable

Normalization

- Reduced data redundancy
- Improved data integrity
- Flexible for expansion
  - Types
    - 1 NF, 2 NF, 3NF, BNCF, 5NF

Solution

StudentID	StudentName
202512434	N1
202512435	N2
202512436	N3
202512437	N4
202512438	N5
202512439	N6

CourseID	CourseName	IsValid	Start Year	End Year
DSC	DataScience	Yes	2020	2035
AIML	Aritificial Intelligence and Machine Learning	Yes	2020	2040
DSCIO	DataScience with IOT	No	2020	2026
DSCML	DataScience and Machine Learning	Yes	2020	2035

SubjectID	SubjectName	Type
SC1	DataScience	Core
SC2	Natural Language Processing	Elective
SC3	Applied Maths	Core
SC4	DataManagement for Machine Learning	Elective
SC5	Advanced communications	Elective

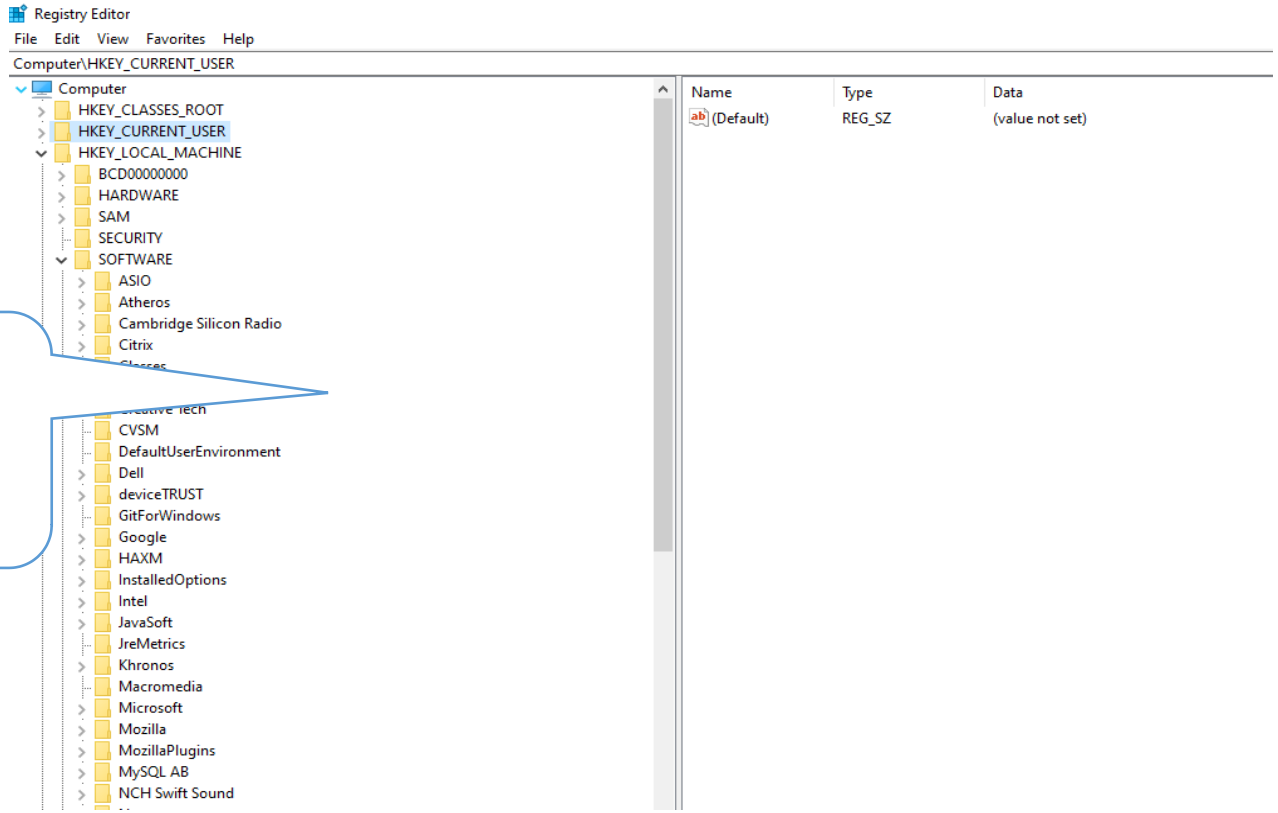
StudentID	CourseID	SubjectID
202512434	DSC	SC1
202512435	DSC	SC1
202512436	DSCIO	SC1
202512437	AIML	SC1
202512438	AIML	SC1
202512439	AIML	SC1
202512434	DSC	SC3
202512435	DSC	SC3
202512436	DSCIO	SC3
202512437	AIML	SC3
202512438	AIML	SC3
202512439	AIML	SC3
202512434	DSC	SC2
202512435	DSC	SC2
202512436	DSCIO	SC4
202512437	AIML	SC4
202512438	AIML	SC5
202512439	AIML	SC5

Hierarchical Data Model

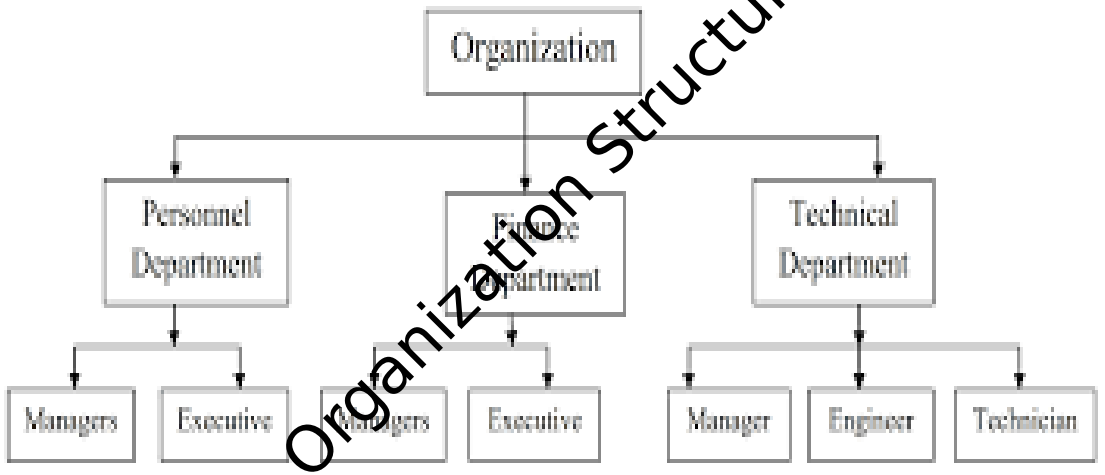
```
<?xml version="1.0"?>
- <birds>
  - <owl id="1201">
      <species>Bubo bubo</species>
      <name>Eagle Owl</name>
      <region>Eurasia</region>
    </owl>
  - <owl id="1202">
      <species>Strix occidentalis</species>
      <name>Spotted Owl</name>
      <region>North America</region>
    </owl>
</birds>
```

Have you seen this window?

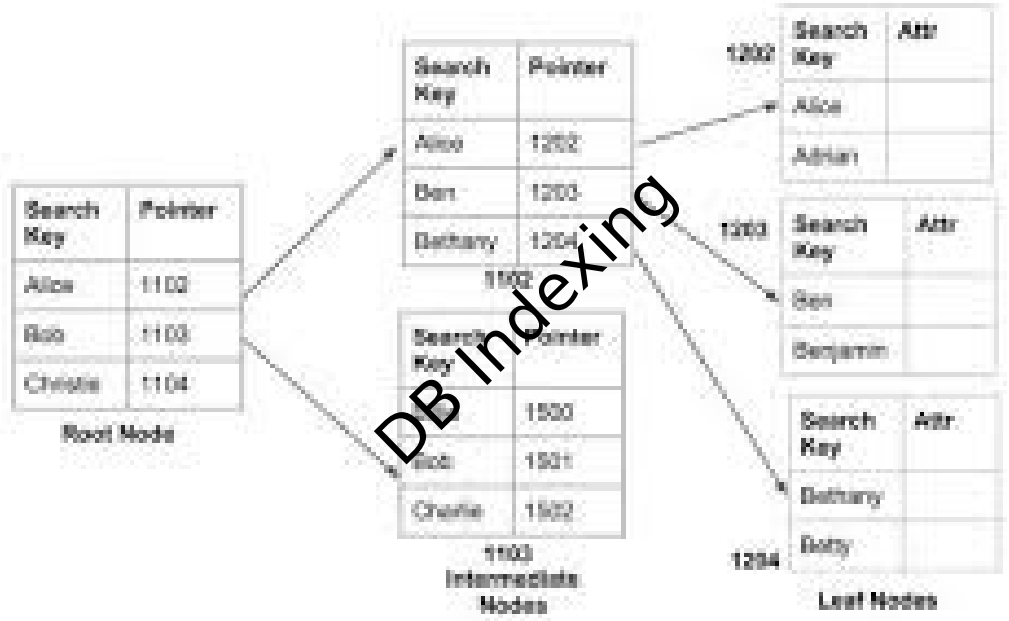
Type Regedit in your command prompt in windows



Where is this used?



Organization Structure



DB Indexing

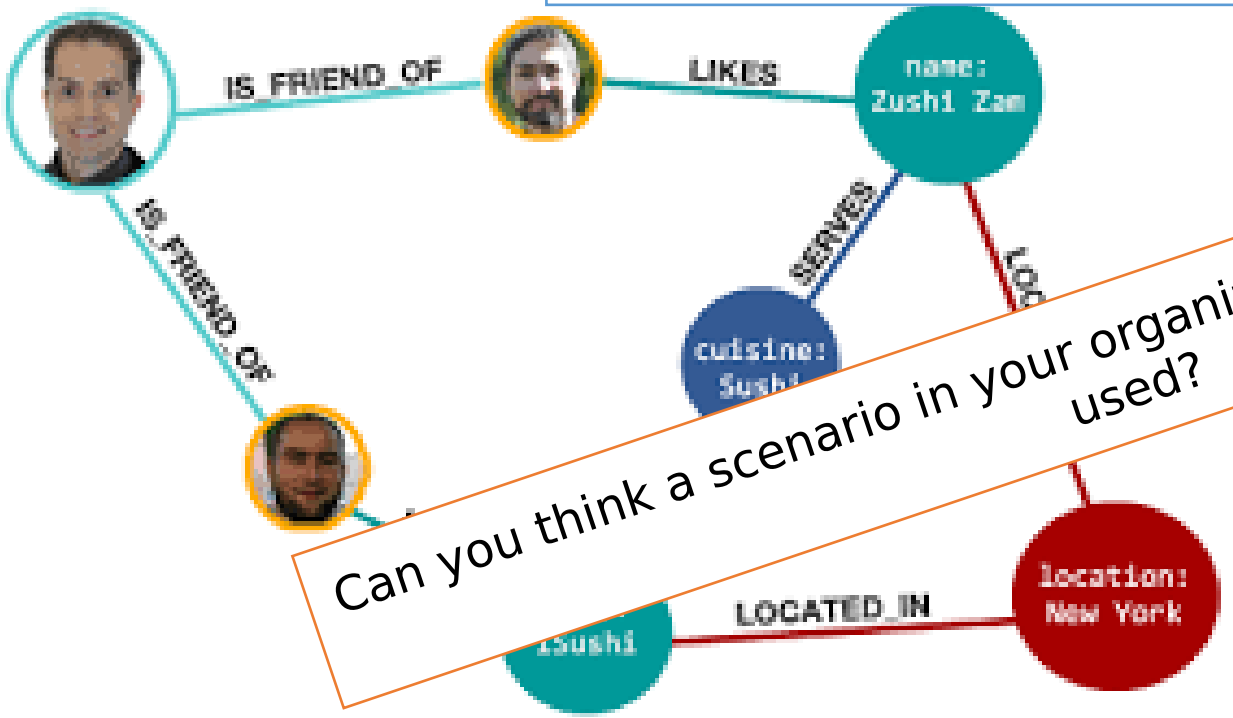
Non clustered index

Graph Datamodel

- Built around concept of “Graph”
- Graph consists of nodes and edges, where edges represents relationship between the nodes
- Database that uses graph structures to store its data is graph database
- In document database, content of each document is priority
- In graph database, relationships between data items are priority!
- Faster to retrieve data faster as relationships are models in that way
- **Eg: Graph Databases : Neo4j, Neptune, Gephi**

Where is this used?

LinkedIn or Facebook



Can you think a scenario in your organization where this is or can be used?

- Sample : Mapping the projects to the business objectives
- to see how many projects are aligned to business objectives
- related to each other in terms of functionality
- redundant with any other in terms of functionality

Document Datamodel

- Built around concept of “document”
- Single continuous string, encoded as JSON, XML
- Each document has a unique key that represents that document, which can be used to retrieve it
- Collection of document analogous to table in relational database, document analogous to row
  - Can convert relation into a collection of documents
  - Collection of document much more flexible than table

For example, a book can be represented as document as follows

```
{
  “Title”: “Book1”,
  “Author” : “Pravin”,
  “Publisher”: “ Press1”,
  “Country”: “US”,
  “Sold as”: [
       { “format”: “Paperback”, “Price”: “20”},
       { “format”: “E-books”, “Price”: “10”},
  ]
}
```

Eg: Amazon DocumentDBApache CouchDB, Firebase Firestore,

Where is this used?

- Storing of unstructured or semi-structured data
- Emails stored as evidences
- Logs from devices or events
- Photos or images for references

Key-Value Pair

Stores data as a collection of key-value pairs

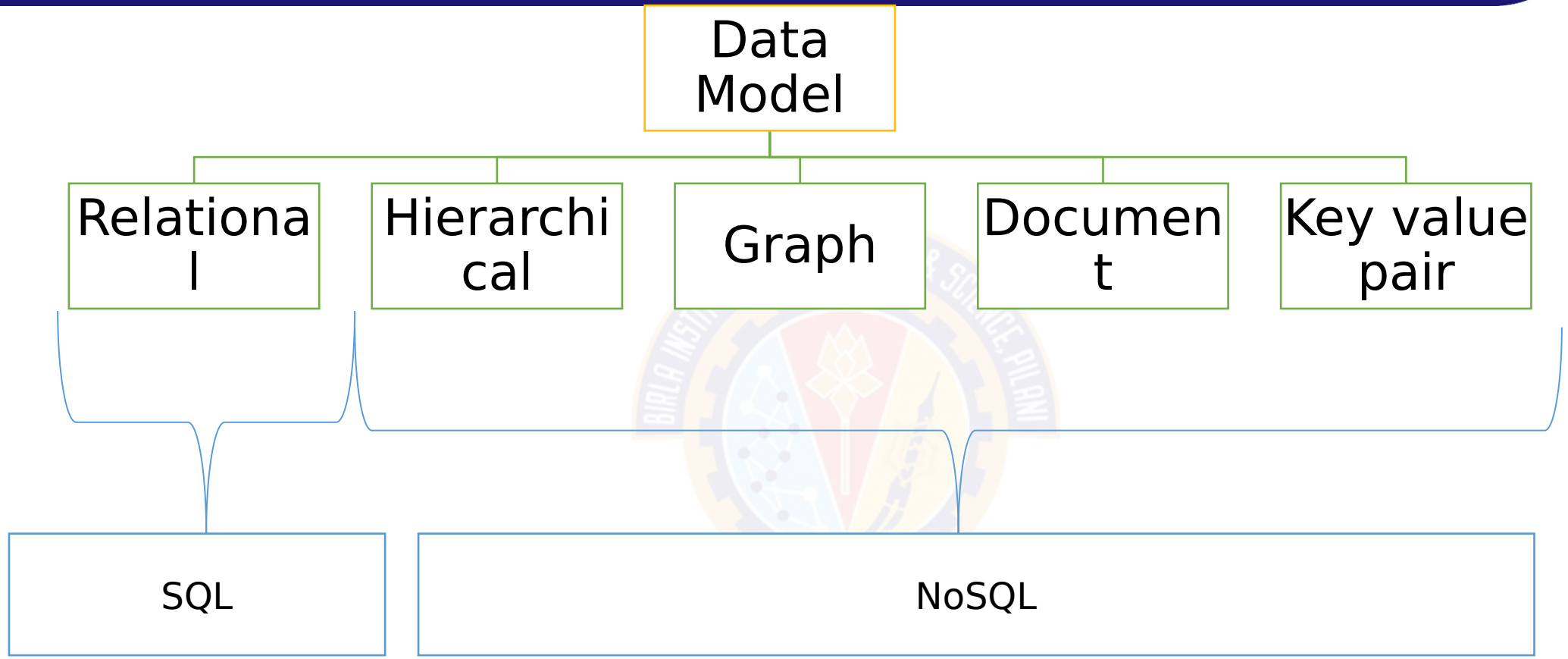
Key	Value
Name	ABC
Subject	DM4ML
Marks	80
Current Status	In progress
Address	no5, street1

Where is this used?

- Caching of data - to **store copies of data in a temporary storage area for faster retrieval**
- JSON objects – internally is a key value pair


```
{
   "user_id_123": {
      "items": [
         {
            "product_id": "P001",
            "name": "Laptop",
            "quantity": 1,
            "price": 1200.00
         },
         {
            "product_id": "P005",
            "name": "Headphones",
            "quantity": 2,
            "price": 150.00
         }
      ]
   },
   "user_id_456": {
      "items": [
         {
            "product_id": "P010",
            "name": "Tablet",
            "quantity": 1,
            "price": 500.00
         }
      ]
   }
}
```

Summary of DataModels

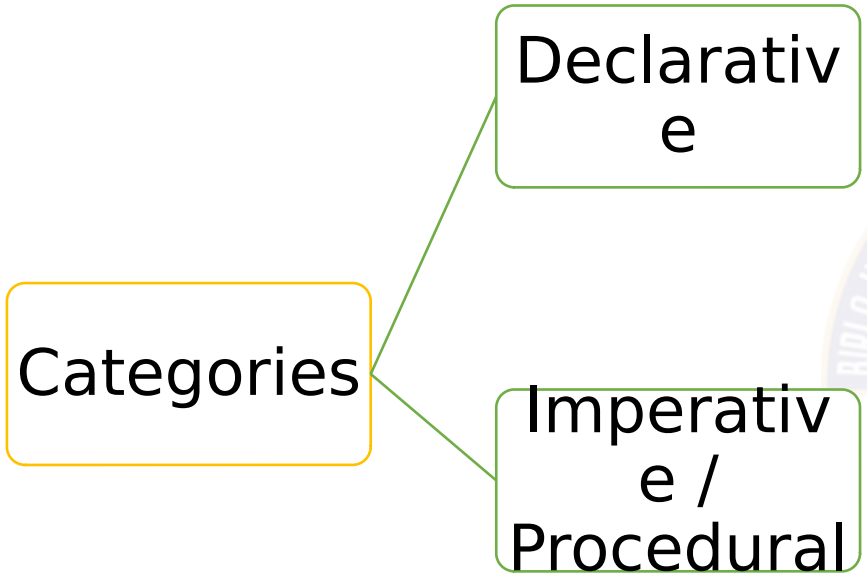


Comparison of various datamodels

	Relational	Hierarchical	Graph	Document	Key-Value Pair
Structure	Strictly schema-based (tables, rows, columns)	Tree-like (parent-child)	Nodes and edges (flexible)	Semi-structured (JSON/XML)	Simple key → value pairs
Querying	SQL (standardized, powerful)	Navigational (e.g., DL/I)	Cypher, Gremlin (relationship-oriented)	JSONPath, MongoDB Query Language	By key only (very limited querying)
Performance (Storage & Retrieval)	Moderate (optimized for complex queries)	Fast for hierarchical traversal	Fast for connected data queries	Fast for document reads/writes	Extremely fast for simple reads/writes
Usage	Traditional business apps, ERP, CRM	Legacy systems, file systems	Social networks, fraud detection, recommendation engines	Content management, catalogs, APIs	Caching, session storage, config data
Sample Databases	MySQL, PostgreSQL, Oracle, SQL Server	IBM IMS	Neo4j, Amazon Neptune, TigerGraph	MongoDB, CouchDB, Amazon DocumentDB	Redis, Amazon DynamoDB, Riak

Query Languages For Data

Basics



- You already have data. Now what next?
- How will you retrieve them and use them next?
- What if you need to retrieve data from multiple tables or databases?
- What is required?
- Let us discuss about query languages

What is the difference?

How these two approaches are different?

Which do you think imperative or declarative?

Why?

Sample:

1. Open the door
2. Go to car parking
3. Open the car
4. Check if the bag is not empty
5. If so, take the bag
6. Close the car
7. Come inside the house
8. Close the door
9. Give the bag to me

Sample:

Bring the bag from the car to me if bag is not empty

Imperative:

Every step of execution should be carefully given

You are detailing the **how** part here

Declarative:

You are just saying what you want

You are asking for **what** you need with all conditions applied

Imperative vs Declarative

Imperative

```
function getSharks() {
    var sharks = [];
    for (var i = 0; i < animals.length; i++) {
        if (animals[i].family === "Sharks") {
            sharks.push(animals[i]);
        }
    }
    return sharks;
}
```

Declarative

```
SELECT * FROM animals WHERE family = 'Sharks';
```

Both wants retrieve sharks from the animals list

- Here the computer is instructed to perform the certain operations in a particular order
  - Stepping through the code
  - Evaluating conditions
  - Updating relevant variables
  - Decision to go and fetch the next record or not
- Here how to achieve the result is given
- The developer/programmer should know the optimal way

- Here just the pattern of data that is needed is specified along with
  - what conditions the results must meet
  - how you want it to be transformed e.g. sorted, grouped and aggregated
- How part is not mentioned
- Up to the database system's query optimizer to decide
- Indexes & joins to use
- Order of execution of various parts of the

Imperative vs Declarative

Imperative

- The underlying system cannot optimize the execution or memory
- Ordering of the output is assured
- Though CPUs are evolving and so an optimal performance can be achieved, still it is explicit mostly
- Parallel execution of a sequence of instruction is quite complex

Declarative

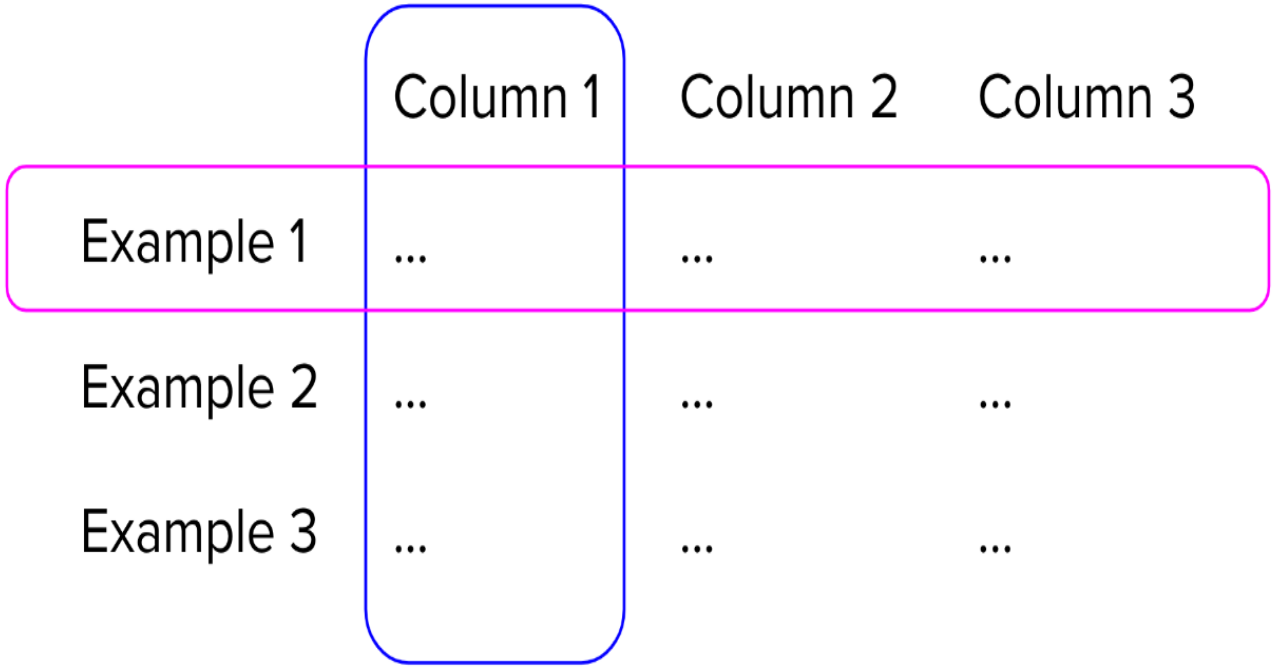
- Easy to work when compared to imperative language
- Hides implementation details
- Database systems can introduce performance improvements with out need for any changes to queries **(Refer to execution plan of SQL for example)**
- Because it is limited in functionality, there are lot of room for the database system to optimize automatically
- Often lend themselves to parallel execution across
- The decision of parallel execution is also

Data Storage Layouts

Row Format vs Column Major Format

Column-major:

- data is stored and retrieved column-by-column
- good for accessing features

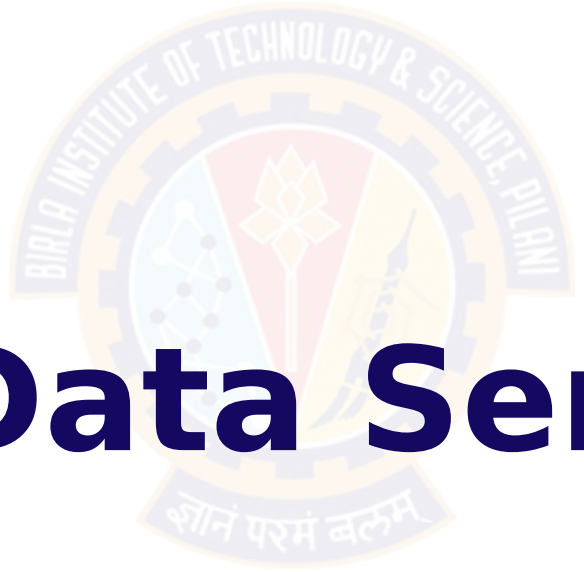


Row-major:

- data is stored and retrieved row-by-row
- good for accessing samples

Row Format vs Column Major Format

Row Format	Column Format
Stores all columns of a row sequentially in the memory	Stores all values of a particular column sequentially in the memory
• If a table is row-major, accessing its rows will be faster than accessing its columns in expectation • For row-major formats, accessing data by rows is expected to be faster than accessing data by columns	• Column-major formats allow flexible column-based reads, especially if data is large with thousands, if not millions, of features
Faster data writes	Faster data reads
Good for OLTP	Good for OLAP
Eg: Format - CSV DB – MS SQL	Eg: Format - Parquet DB – Amazon Redshift

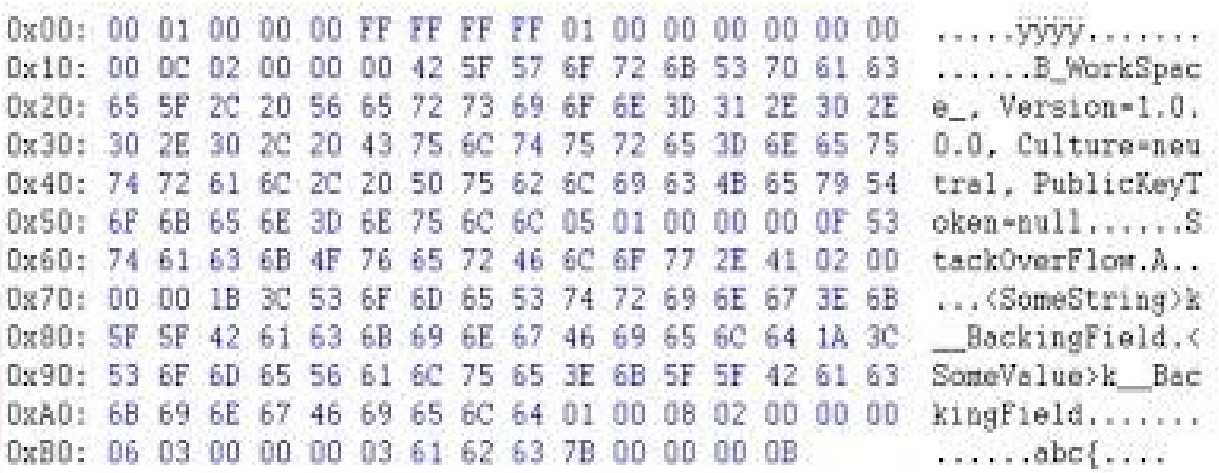


Data Serialization

What and Why?

What?

- In Simple terms – it is converting of one data from one form to other
- Example converting from text to binary format



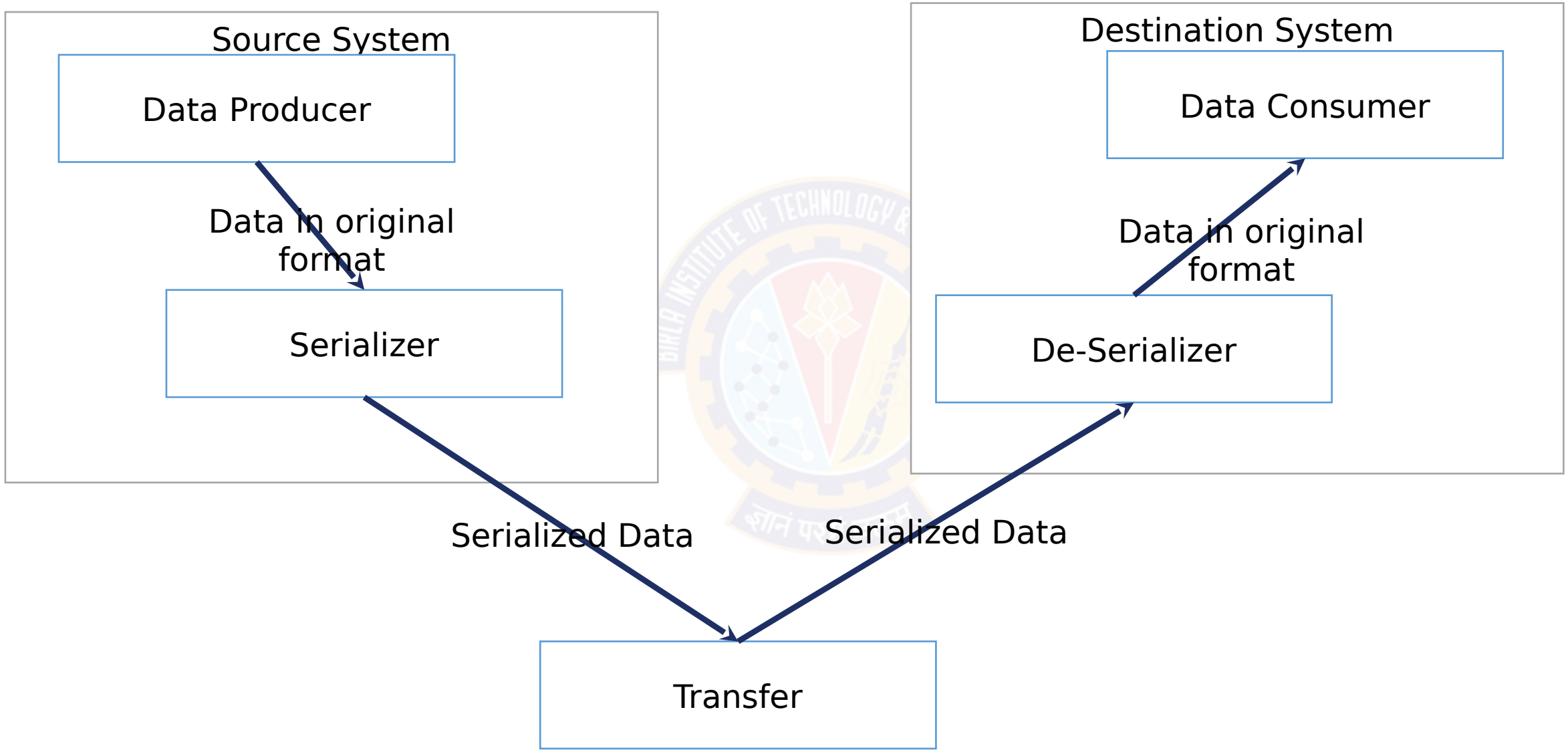
Why?

- To easily
  - Store – size constraints
  - Transfer – bandwidth constraints

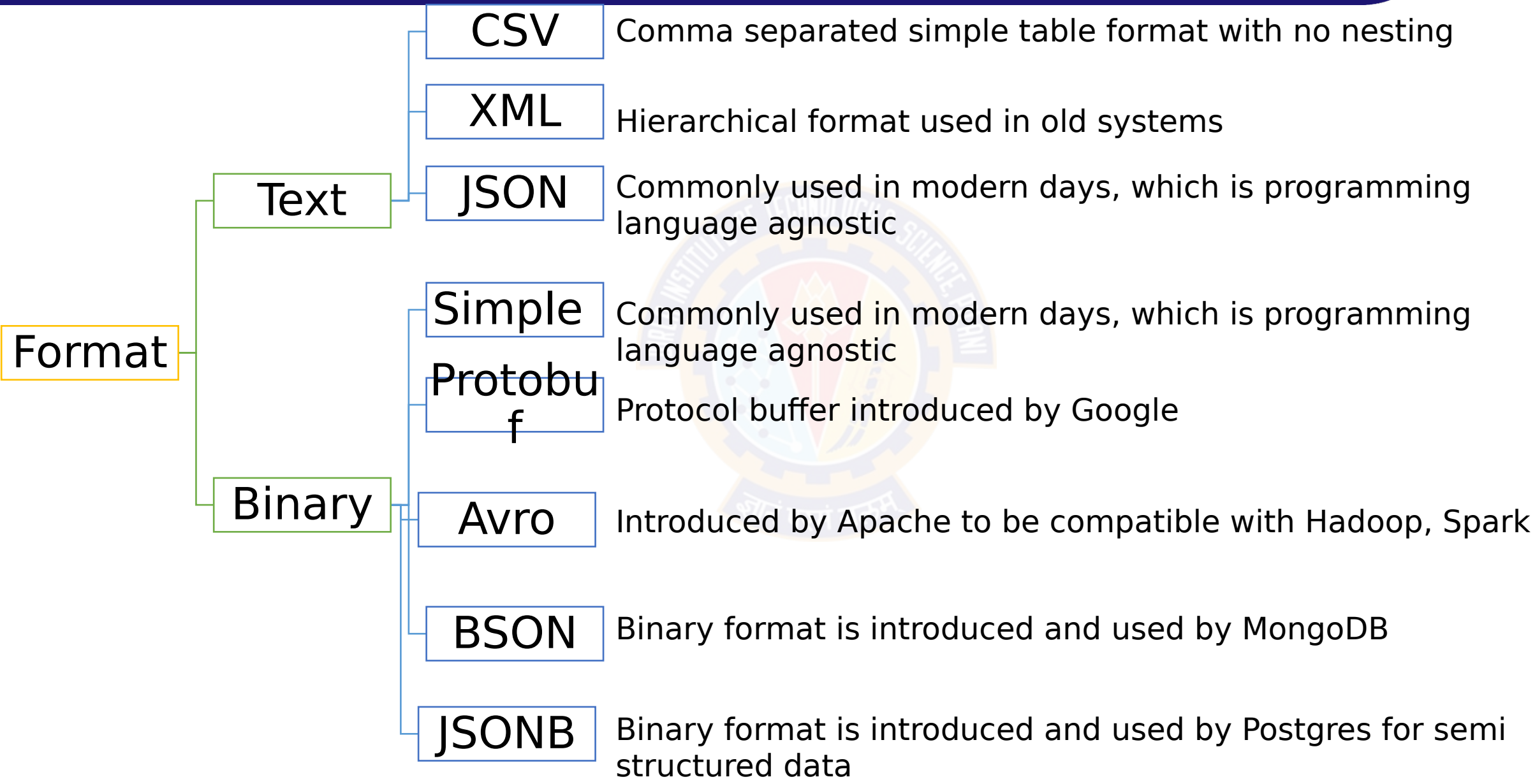
Points to Remember

- Both serialization & deserialization should use same algorithm
- Care should be taken if different programming languages or technologies used during the

Process


Source System
Data Producer
Data in original format
Serializer
Serialized Data
Destination System
Data Consumer
Data in original format
De-Serializer
Serialized Data
Transfer

Serialization Formats


Format
Text
CSV
Comma separated simple table format with no nesting
XML
Hierarchical format used in old systems
JSON
Commonly used in modern days, which is programming language agnostic
Binary
Simple
Commonly used in modern days, which is programming language agnostic
Protobuf
Protocol buffer introduced by Google
Avro
Introduced by Apache to be compatible with Hadoop, Spark
BSON
Binary format is introduced and used by MongoDB
JSONB
Binary format is introduced and used by Postgres for semi structured data

Sample

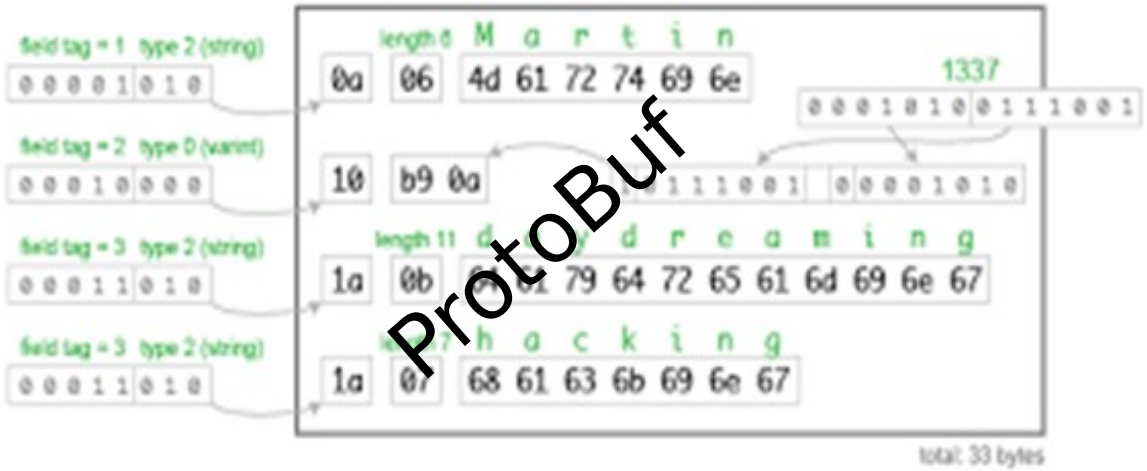
CSV

```
amphitheaters.csv - Notepad
File  Edit  Format  View  Help
"Roman","Modern","Country","Year","Length","Notes","Photo","Latitude","Longi
"Dyrrhachium","Durrës","Albania","2nd century AD","61 m","Durrës Amphitheatr
"Lambaesis","Lambèse","Algeria",,"64 m",,,,35.489247,6.259935
"Colonia Claudia Caesarea","Cherchell","Algeria",,"93 m",,,,36.60874,2.
"Gemellae","M'Lili","Algeria",,"? m",,,,34.635409,5.522764
"Theveste","Tébessa","Algeria","4th century AD","45 m","Aerial Photograph",
"Tipasa","Tipaza","Algeria",,,"Map of Tipasa","https://en.wikipedia.org/
"Carnuntum","Petronell","Austria",,"69 m","2 amphitheatres","https://en.
"Carnuntum","Petronell","Austria",,"69 m","2 amphitheatres","https://en.
"Flavia Solva","Leibnitz","Austria",,,,,,46.766744,15.567417
"Virunum","Magdalensberg","Austria",,,,,"https://en.wikipedia.org/wiki/
"Diocletianopolis","Hisarya","Bulgaria",,,,,,42.902825,24.709776
"Marcianopolis","Devnya","Bulgaria",,,,,,43.222222,27.569444
"Serdica","Sofia","Bulgaria","3rd century AD",,"In ground floor of Arena c
"Pietas Julia Pola","Pula","Croatia","1st century AD","68 m","Pula Arena","
"Salonae","Solin","Croatia",,"65 m",,"https://en.wikipedia.org/wiki/Fils
"Burnum",,"Croatia",,"46 m","Roman military camp near Sibenik, had a sma
"Augusta Paphus","Paphos","Cyprus",,"65 m",,,,34.754942,32.405344
"Kolossi",,"Cyprus",,"Amphitheatre almost destroyed",,,35.695533,33.
```

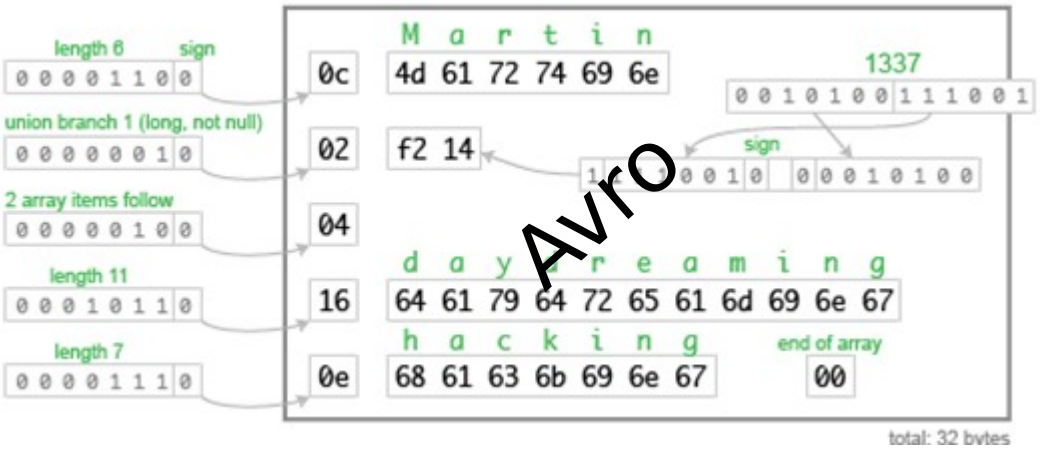
JSON

```
{
  "api": {
    "title": "Example API",
    "links": {
      "author": "mailto:api-admin@example.com",
      "describedBy": "https://example.com/api-docs/"
    }
  },
  "resources": {
    "tag:me@example.com,2013:widgets": {
      "href": "/widgets/"
    },
    "tag:me@example.com,2013:widget": {
      "hrefTemplate": "/widgets/{widget_id}",
      "hrefVars": {
        "widget_id": "https://example.org/param/widget"
      },
      "hints": {
        "allow": ["GET", "PUT", "DELETE", "PATCH"],
        "formats": {
          "application/json": {}
        },
        "acceptPatch": ["application/json-patch+json"],
        "acceptRanges": ["bytes"]
      }
    }
  }
}
```

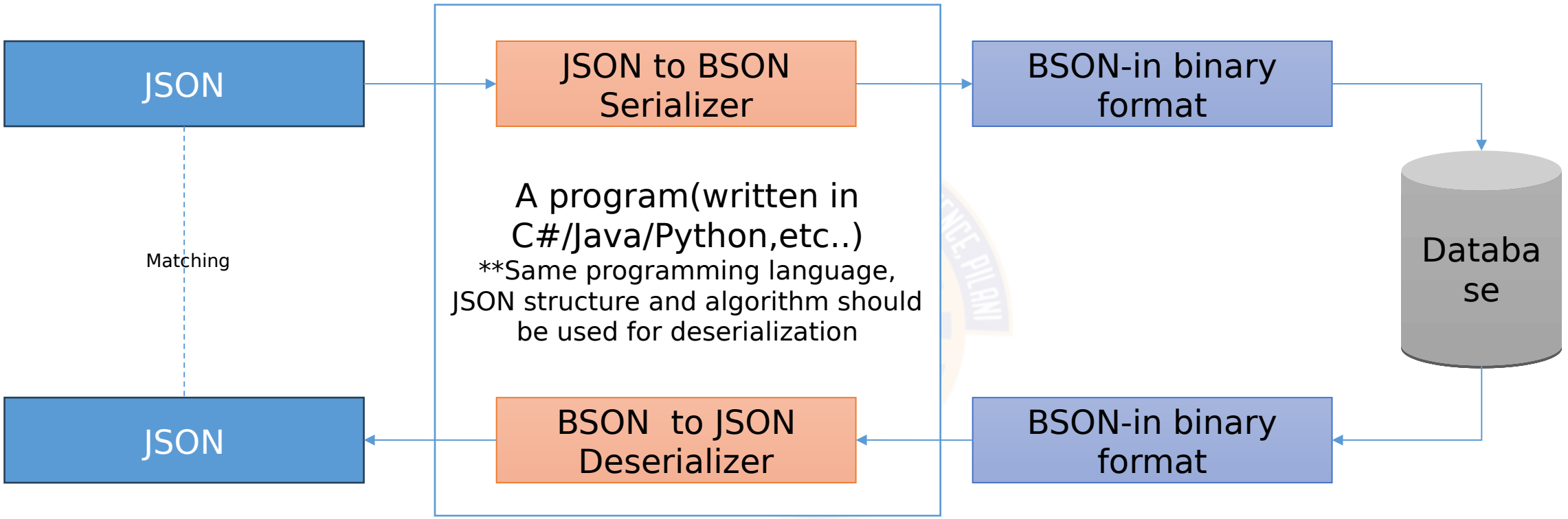
ProtoBuf


field tag = 1 type 2 (string)
0 0 0 0 1 0 1 0
length 6
M a r t i n
0a 06 4d 61 72 74 69 6e
1337
0 0 0 1 0 1 0 0 1 1 1 0 0 1
field tag = 2 type 0 (varint)
0 0 0 1 0 0 0 0
10 b9 0a
field tag = 3 type 2 (string)
0 0 0 1 1 0 1 0
length 11
d a y d r e a m i n g
1a 0b 64 61 79 64 72 65 61 6d 69 6e 67
field tag = 3 type 2 (string)
0 0 0 1 1 0 1 0
h a c k i n g
1a 07 68 61 63 6b 69 6e 67
total: 33 bytes

Avro



Sample Process



This is a 2 step process viz.,

1. JSON is converted to BSON format
2. BSON is converted to binary format

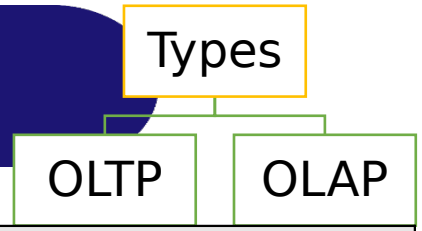
Summary

Format	Binary/Text	Human-readable	Example use cases
JSON	Text	Yes	Everywhere
CSV	Text	Yes	Everywhere
Parquet	Binary	No	Hadoop, Amazon Redshift
Avro	Binary primary	No	Hadoop
Protobuf	Binary primary	No	Google, TensorFlow (TFRecord)
Pickle	Binary	No	Python, PyTorch serialization

Common data formats and where they are used.

Data Processing Types

Data Processing Types



	OLTP	OLAP
Stands for	Online Transaction Processing	Online Analytical Processing
Purpose	To save the transactions immediately	Data derived from OLTP for analytical purposes
Type of data	Current, operational data	Historical, consolidated data
Volume	Handles many small transactions	Handles fewer but large data sets
Optimization	For faster write	For faster reads
Usage example	• Gpay transactions • Swipe records of your entry and exit in your office	• Gpay analysis on your spend • Attendance calculation with the swipe record

OLAP – For insights

Sharp enterprise-wide profitability dip

Sales OK, profitability down last 3 months

Sharp reduction in European region

Increase in a few countries, flat in others, sharp decline in some

Sharp decline in EU countries, last 2 months

Direct costs OK, indirect costs up

Additional tax on some products in EU

Countrywide monthly sales for last 3 months?

Monthly sales by worldwide regions?

European sales by countries ?

European sales by countries, products?

Direct and indirect costs for EU countries?

Typical Calculations

- Roll-ups to provide summaries and aggregations along the hierarchies of the dimensions.
- Drill-downs from the top level to the lowest along the hierarchies of the dimensions,
- in combinations among the dimensions.
- Simple calculations, such as computation of margins (sales minus costs).
- Share calculations to compute the percentage of parts to the whole.
- Algebraic equations involving key performance indicators.
- Moving averages and growth percentages.

OLAP operations

LINE	TOTAL SALES
Clothing	$12,836,450
Electronics	$16,068,300
Video	$21,262,190
Kitchen	$17,704,400
Appliances	$19,600,800
Total	$87,472,140

Drill down

LINE	1998	1999	2000	TOTAL
Clothing	$3,457,000	$3,590,050	$5,789,400	$12,836,450
Electronics	$5,894,800	$4,078,900	$6,094,600	$16,068,300
Video	$7,198,700	$6,057,890	$8,005,600	$21,262,190
Kitchen	$4,875,400	$5,894,500	$6,934,500	$17,704,400
Appliances	$5,947,300	$6,104,500	$7,549,000	$19,600,800
Total	$27,373,200	$25,725,840	$34,373,100	$87,472,140

Rotate / Pivot

YEAR	Clothing	Electronics	Video	Kitchen	Appliances	TOTAL
1998	$3,457,000	$5,894,800	$7,198,700	$4,875,400	$5,947,300	$27,373,200
1999	$3,590,050	$4,078,900	$6,057,890	$5,894,500	$6,104,500	$25,725,840
2000	$5,789,400	$6,094,600	$8,005,600	$6,934,500	$7,549,000	$34,373,100
Total	$12,836,450	$16,068,300	$21,262,190	$17,704,400	$19,600,800	$87,472,140

Features of OLAP

Multidimensional analysis	**Consistent performance**	**Fast response times for interactive queries**
Drill-down and roll-up	**Navigation in and out of details**	**Slice-and-dice or rotation**
Multiple view modes	**Easy scalability**	**Time intelligence (yearto-date, fiscal period)**

Basic Features

Advanced Features

Powerful calculations	**Cross-dimensional calculations**	**Pre-calculation or pre-consolidation**
Drill-through across dimensions or details	**Sophisticated presentation & displays**	**Collaborative decision making**
Derived data values through formulas	**Application of alert technology**	**Report generation with agent technology**

References:

- Designing Data-Intensive Applications by Martin Kleppmann
- Reliable Machine Learning By Chen, Murphy, Sculley, Underwood
- Designing Machine Learning Systems by Chip Huyen
- https://phoenixnap.com/kb/acid-vs-base

Thank You!

In our next session: